GENCODE: PREPARATION FOR THE FUTURE

Aaron H. Bigman, Program Director, Information Processing, Graphic Communications Association

Abstract: In recent years, the concept of generic text element identification, or generic coding, has attracted the interest of the "text processing" (printers, publishers) and information utility industries. The Graphic Communications Association has been active in supporting and sponsoring generic coding (GenCode) development and is most recently conducting "benchmarks" and tests to evaluate and enhance the concept into a set of guidelines. Moreover, a number of large companies are using generic coding schemes as a generalized markup language (GML) for their text processing requirements. As more companies seriously begin to examine interfaces between word processors and typesetters, the generic coding concept attracts more attention. The reasons for this attraction will be presented in the following paragraphs.

The concept of generic coding is quite simply, borrowed from a basic tenent of data base management thoery: "Separate the data from the application." In text processing environments, the data becomes the intellectual content, or text, and the application is that of format; how the text (and graphics) will be visually presented, whether on paper or through an electronic display. Formatting is therefore that method that supports the intellectual content of the material and renders it "readable."

The beauty of a descriptive and sequential generic coding scheme is that it lends itself to the inherent logical structure of a document. Text elements, by their very nature, demonstrate a hierarchial structure. For example, a book is composed of zones, such as front matter, body matter and back matter. Each of these zones has segments. Front matter may consist of an appendix and an index. The generic coding of these elements, irregardless of format considerations, is therefore a logical and simplified approach to generalized markup.

Generic coding is, and can be, of benefit to those generating information products through the use of automated text processing equipment. Data entry costs are much lower, text files are device independent and void of any unique markup, and can be manipulated at will. Suppliers and consumers of these information products will also benefit indirectly. Equipment suppliers will benefit from the enhanced ability to serve a broader array of customers and increased market opportunities. Authors will benefit from greater control over their product, and enhanced acceptability of their manuscript from a wider selection of publishers. Publishers will also accrue benefits through the multiple uses of their data bases, and greater vendor independence.

It is now quite obvious to those of us in the publishing and printing industry that the 1980's will offer a variety of print and non-print options for the delivery of information. This conference is quite actively promoting the non-print, electronic delivery option, where the publisher electronically disseminates the product to the consumer. A project is now underway in Europe to provide a variety of electronic outlets for publishers of scientific, medical and technical publications. This system should by 1984, utilize satellite, PBX and local area networks for the delivery of data in unstructured or page image formats throughout Europe.

A central issue is the generation of this information. It is being done now, today, with the use of those devices, ranging from pencil and paper, typewriters, word processors to data terminals. There is however, a noticeable evolution of authors to electronic date capture devices. A study conducted in England last year discovered that 60% of the authors of primary journals were using an electronic data capture device of some type. This obviously facilitates text creation and editing, provides a soft storable copy readily updatable, and can potentially become the delivery vehicle to the publisher. Most American publishers recognize that by capturing the author's keystrokes, the production cycle can be significantly shortened. A faster cycle offers marketing advantages, as well as a lower production cost. Capturing the author's keystrokes saves time and money by eliminating data entry costs for photocomposition, eliminates proofreading and provides an "interactive" manuscript between the author and editor.

The author works, however, in a world full of incompatabilities. The author's electronic data capture device, for example, a word processor, has its own unique and non-standard formatting and control codes that style the manuscript. Publishers usually do not have any control over what type of device the author uses. Therefore, to permit each author to electronically generate a manuscript and deliver, in a machine sense, that copy, the publisher would theoretically have to build compatability with every conceivable type of device. And, moreover, with the various print and non-print options for the delivery, what processing terminology (format/control coding) can the publisher request that the author use? How can the publisher technically prepare his data files for a variety of print and non-print delivery devices? The questions have various solutions in a processing environment (code translations), but lets look again at the author. The author's intent is to create an intellectual object; a book or an article that is more or less the "mechanical" of the creative thinking process. Most authors therefore resist any "keystroking" that does not achieve the initial objective of capturing the creative thought. They resist imparting typographic codes in the data files. But if they don't, the publisher or compositor has to read the implicit and explicit source specific codes and reconstruct, for his processing equipment, the data stream.

A "today" solution appears to the GCA's GenCode; our term for a powerful generic coding language. Generic coding is a concept which incorporates a definition of the intellectual content of data into a file in machine processable form. Generic coding is a concept, not a programming language. It is not even a new concept. It is more a new phrase being used relative to text processing than an original idea. If you think about it, generic coding is a traditional computer coding scheme for non-text applications. When a programmer says positions 1 to 10 of a record contain the part number of a product, and positions 11 to 15 are the price, he is really doing generic coding - incorporating a definition of the intellectual content of the data into the file by giving a meaning to the location of the data

within each record. This form of coding is considered to be implicit coding because no actual code exists in the file to impart the meaning of the data.

In the case of running text files, implicit coding does not work since there is no fixed location within a record that can be assigned a generic meaning. In running text files, some sort of embedded explicit code must be used to give a meaning to the data being processed.

GCA's GenCode Committee, consisting of GCA members from various segments of the publishing and printing industry, are currently completing their work on the refinement of the GenCode concept into a language, and thus, a syntax that an author can use for the creation of a manuscript. Let's see how it can work. The capture and electronic manipulation of a manuscript for processing results in the capture of two types of information. One is the "text," the intellectual structure, usually organized in outline form, and then enhanced by the use of words and symbols for the content. The second type of information which is captured is instructional; information that facilitates and instructs the device on the handling, management and manipulation of the intellectual content. In a systems environment, interface is accomplished by having the author and the publisher agree on a standard set of conventions for both content and control information. In terms of content, most processing systems recognize the standard character sets for alphanumerics. The machine creation and understanding of alphanumeric information (the text, or intellectual context) can therefore be established by the user of these standards. The first issue to be resolved, however, is with the "extended" character set. These elements consist of non-alphanumeric characters that have not been recognized and standardized to date. (Although the AT&T Videotext standard for this, and character attribute information needs study in this context.) An approach for the recognized recording and transmission of these non-alphanumerics should focus upon using, however, the ASCII keyboard for reasons of availability. Diacritic markings can be used to supplement the keyboard, as is being done in proto-type development for mathematical equations, chemical formulas, symbols and graphics. GCA is monitoring these developments quite closely, since the goal is really total document capture and transmission with text and non-textual elements being keyboard by anyone with access to an ASCII keyboard. This also has an impact on character coding, since codes for these symbols on processing systems are totally source specific.

A more difficult problem arises with the use of instructional information. This information is primarily formatting codes of various types; influencing either the positioning of elements, page formatting or special control features for character and element presentation. The formatting codes of word processing equipment and composition programs is non-standardized. Therefore, there is a great deal of system incompatability in this area. Unfortunately, these problems are being resolved in source specific way through the development and application of hardware and software solutions to strip, read and translate codes. In a non-controlled publishing environment, the publisher (being the receiver) would have to develop comability with any and every conceivable device. Moreover, in-house composition and data manipulation practices would be without a standard and structured approach to total text management. Also, format-oriented coding lacks the intelligence that is required for more sophisticated, non-typographic applications, such as for data base development. Elaboration of this problem has been done elsewhere; the point is that standardization begins at the elimination of these instructional codes.

An intelligent approach to instructional coding is through a markup that is content and not format oriented. Content oriented codes are

commonly called generic markup languages (GML). GCA's approach to GML, GenCode, implements a coding (markup) approach that is oriented to the intellectual content, and therefore, format independence. The type of markup used by an author would be instructional only in its relation to the manuscript and its elements; capturing, in a machine sense, the structure and hierarchy of the document elements; (parts, sections, chapters, heads, paragraphs, citations, footnotes, tables, illustrations, etc.). These elements exist in a document on horizontal (equal) and vertical (subordinate) levels. By separating the data, which is the elements in a manuscript, from the application, how those elements will be formatted, a GenCoded manuscript places minimal coding responsibility upon the author (nothing other than the electronic capture of his outline and the identification of specific elements) and places composition and formatting with the publisher. Authors, additionally, should not be required to perform format-oriented coding; most, infact, have resisted and are discouraged by it. Studies have shown, that, on the other hand, a GML approach, is simple, logical and structually sound as (hopefully) is the manuscript outline.

A GML is, again, not a processing code, but instructional information in a machine processing environment that relates the document structure, its elements, the interrelationship of those elements and their position. Sophisticated GML approaches involve the creation of standard document types which map directly to format processing. The advantage, though, is that the author creates a totally device independent "electronic" manuscript. The result is an intelligent source file that can be exploited for both print and non-print applications.

Electronically interfacing with an author therefore, beings with the publisher's issuance of a handbook which details GML, suggests the generic identifiers to be used, delimiters to distinguish between markup and content, and extentions for the character set. With this data, an author can use any type of word processor or terminal, and deliver readible copy with intelligent markup to the publisher. Such handbooks are currently being devised. Thus, there is now a wealth of information that the book publishing production community can draw upon to "interface" with their authors -- with no capital investment in translation/code conversion equipment. For the authors, anyone with a standard ASCII keyboard can create full documents including text, non-textual elements and instructional information. Much work still needs to be done toward the accomplishment of this last goal, but importantly, the approach has been used, tested and validated.

Upon the inheritance of a GML file (communication issues will not be addressed in this paper, although they obviously have both technical and cost implications) the publisher then begins the composition and makeup function. The inheritance consists of a virtual scroll of both markup and content information; no line, page breaks or other format comands. An assumption is that the marketing and sales staff, working with the graphic arts designers, have developed a profile for the individual publication. This profile consists of both structural and format conditions. Therefore, the technical step is to "qualify" the generic identifiers in the author's GML file to valid, source-specific display or print commands. A page makeup station, consisting, let's assume, of a bit map display device is used for page composition. The qualification of the generic identifiers, and the interpretation for the extended characters then displays the text of the document in the desired format on the screen for copy editing. A bit map display (such has attracted my imagination) displays not only the composition element format, but the type face, point size and associated attributes as well. Page make is thus performed interactively, integrating illustra-

tions and graphics where and if necessary. A screen dump then outputs to a high resolution laser printer for proofing, and after copy or design corrections are made, the GML file is "requalified" for typesetting.

The process of qualification is the actual processing of a GML file. The generic tags are assigned processing attributes (source specific coding values) depending on the machine code of the display or output device. The tags can be qualified per application on a one for one basis, or mapped against a look-up table for each different document type and format.

By employing a GenCode structure to the electronic manuscript creation process, the author's intellectual content is captured not only in a device-independent sense, but in a process-independent sense as well. The resultant data file is as applicable to print delivery system as it is to non-print delivery systems. The author intellectually creates the manuscript and describes, through a content-oriented markup, its structure and data elements. The author does not have to be concerned with processing (from a production sense). The publisher saves time and money by capturing the author's keystrokes, and inherits a machine-processable data file that is suitable for a variety of product and delivery applications. Thus, future delivery vehicles of the publisher's product need not, technically, be anticipated.